Wine_Quality_Prediction

April 30, 2023

1 WINE QUALITY PREDICTION

Importing The Libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

Read the csv file into a DataFrame

[3]:
```
df=pd.read_csv('wine.csv')
```

[4]:
```
df.head(10)
```

[4]:
```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076
5            7.4              0.66         0.00             1.8      0.075
6            7.9              0.60         0.06             1.6      0.069
7            7.3              0.65         0.00             1.2      0.065
8            7.8              0.58         0.02             2.0      0.073
9            7.5              0.50         0.36             6.1      0.071

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56
5                 13.0                  40.0   0.9978  3.51       0.56
```

```
6                 15.0                    59.0   0.9964  3.30       0.46
7                 15.0                    21.0   0.9946  3.39       0.47
8                  9.0                    18.0   0.9968  3.36       0.57
9                 17.0                   102.0   0.9978  3.35       0.80

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
5      9.4        5
6      9.4        5
7     10.0        7
8      9.5        7
9     10.5        5
```

[5]:
```
df.shape
```

[5]:
```
(1599, 12)
```

[6]:
```
df.isnull().sum()
```

[6]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

[8]:
```
df.describe()
```

[8]:
```
       fixed acidity  volatile acidity  citric acid  residual sugar  \
count    1599.000000       1599.000000  1599.000000     1599.000000
mean        8.319637          0.527821     0.270976        2.538806
std         1.741096          0.179060     0.194801        1.409928
min         4.600000          0.120000     0.000000        0.900000
25%         7.100000          0.390000     0.090000        1.900000
50%         7.900000          0.520000     0.260000        2.200000
75%         9.200000          0.640000     0.420000        2.600000
```

```
max       15.900000          1.580000     1.000000       15.500000

          chlorides  free sulfur dioxide  total sulfur dioxide      density  \
count   1599.000000          1599.000000           1599.000000  1599.000000
mean       0.087467            15.874922             46.467792     0.996747
std        0.047065            10.460157             32.895324     0.001887
min        0.012000             1.000000              6.000000     0.990070
25%        0.070000             7.000000             22.000000     0.995600
50%        0.079000            14.000000             38.000000     0.996750
75%        0.090000            21.000000             62.000000     0.997835
max        0.611000            72.000000            289.000000     1.003690

                 pH    sulphates      alcohol      quality
count   1599.000000  1599.000000  1599.000000  1599.000000
mean       3.311113     0.658149    10.422983     5.636023
std        0.154386     0.169507     1.065668     0.807569
min        2.740000     0.330000     8.400000     3.000000
25%        3.210000     0.550000     9.500000     5.000000
50%        3.310000     0.620000    10.200000     6.000000
75%        3.400000     0.730000    11.100000     6.000000
max        4.010000     2.000000    14.900000     8.000000
```

[9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

[10]:

```
df.dtypes
```

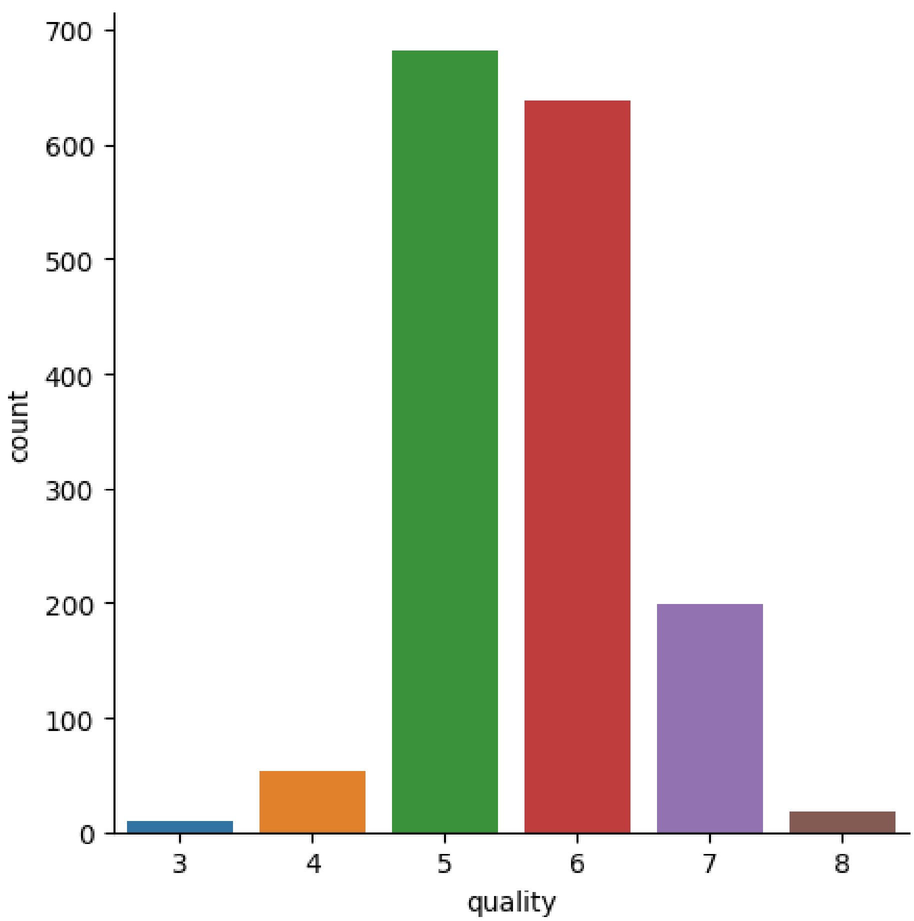
[10]:
```
fixed acidity           float64
volatile acidity        float64
citric acid             float64
residual sugar          float64
chlorides               float64
free sulfur dioxide     float64
total sulfur dioxide    float64
density                 float64
pH                      float64
sulphates               float64
alcohol                 float64
quality                   int64
dtype: object
```

Visualization of Data

Number of values for each quality

[16]:
```
sns.catplot(x='quality', data = df, kind = 'count')
```

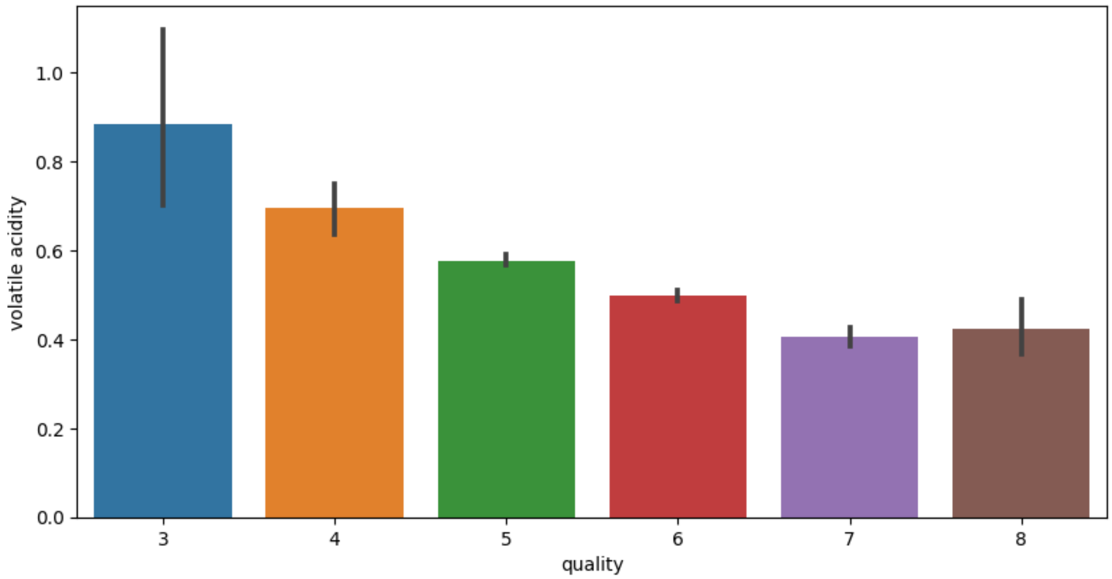
[16]:
```
<seaborn.axisgrid.FacetGrid at 0x7feb6a606a70>
```



Volatile Acidity vs Quality

[18]:
```
plot = plt.figure(figsize=(10,5))
sns.barplot(x='quality', y = 'volatile acidity', data = df)
```

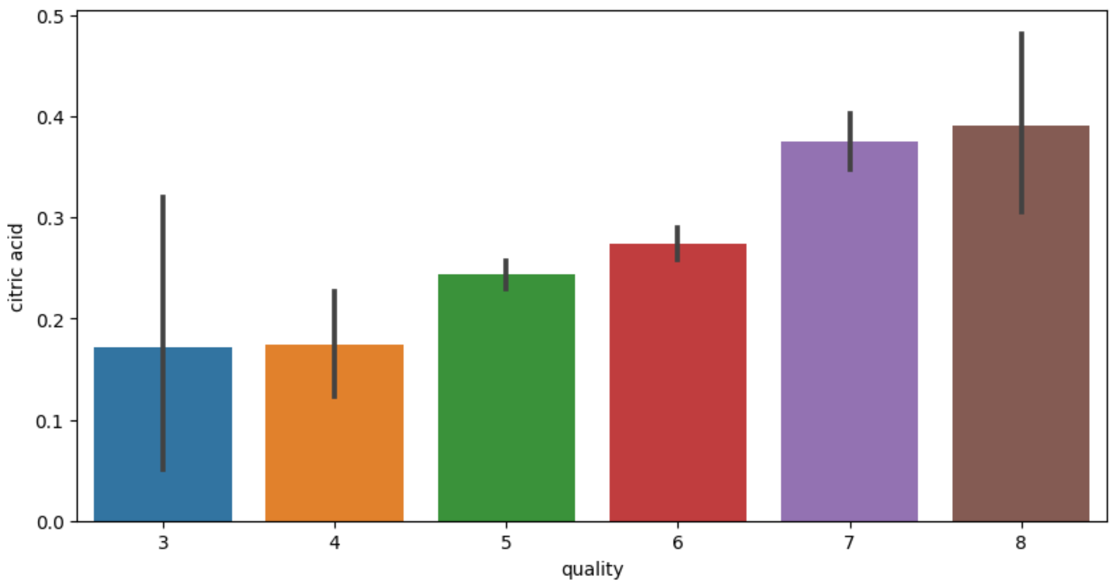
[18]: <Axes: xlabel='quality', ylabel='volatile acidity'>



Citric Acid vs Quality

[19]:
```
plot = plt.figure(figsize=(10,5))
sns.barplot(x='quality', y = 'citric acid', data = df)
```

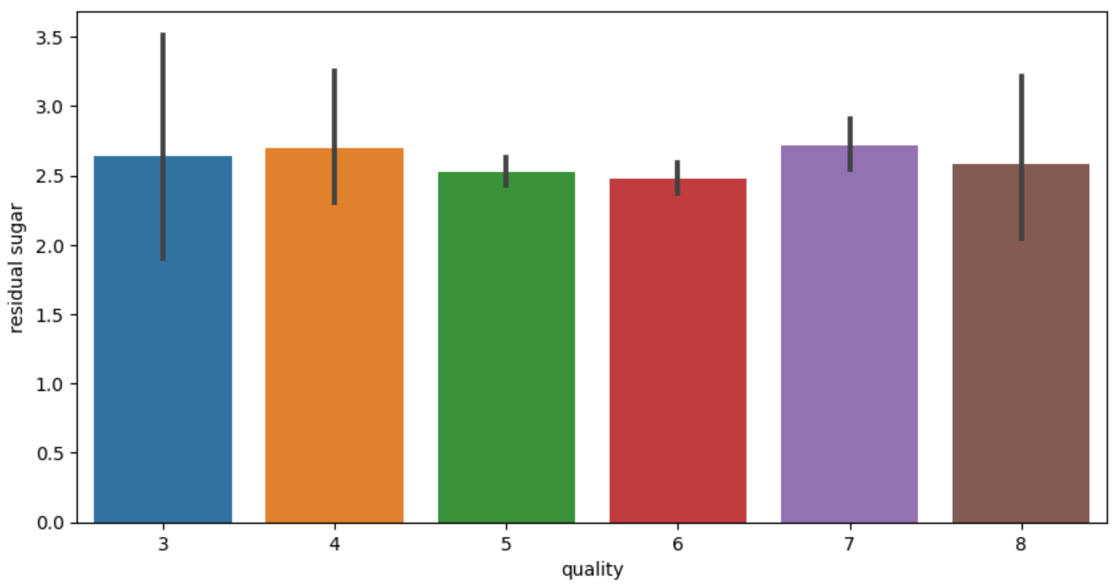
[19]: <Axes: xlabel='quality', ylabel='citric acid'>



Residual Sugar vs Quality

[20]:
```
plot = plt.figure(figsize = (10,5))
sns.barplot(x = 'quality', y = 'residual sugar', data = df)
```

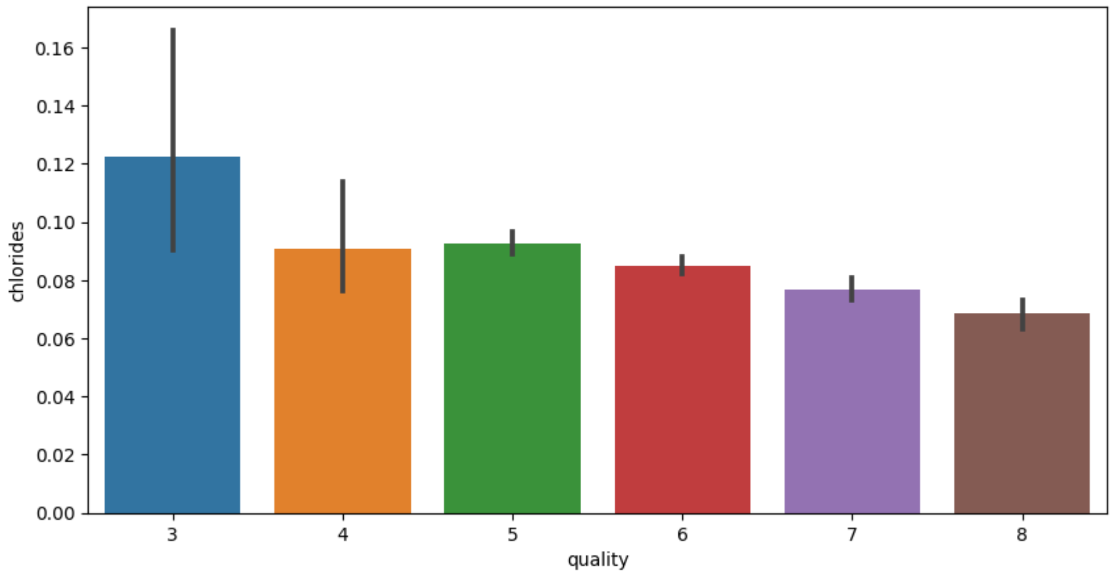
[20]: <Axes: xlabel='quality', ylabel='residual sugar'>



Chlorides vs Quality

[22]:
```
fig = plt.figure(figsize = (10,5))
sns.barplot(x = 'quality', y = 'chlorides', data = df)
#Composition of chloride also go down as we go higher in the quality of the wine
```

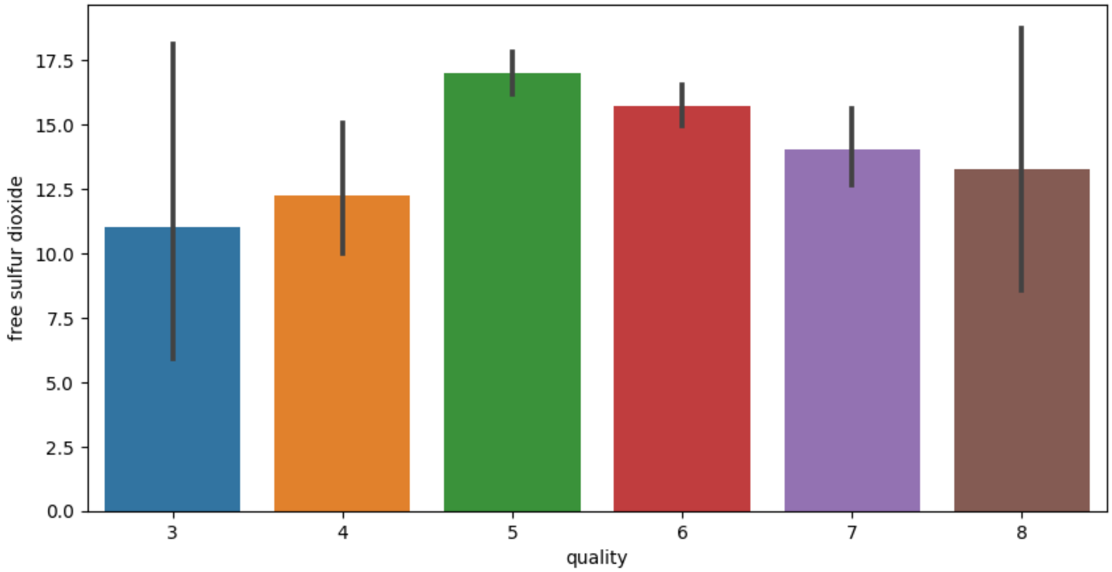
[22]: <Axes: xlabel='quality', ylabel='chlorides'>



Free Sulphur Dioxide vs Quality

[23]:
```
fig = plt.figure(figsize = (10,5))
sns.barplot(x = 'quality', y = 'free sulfur dioxide', data = df)
```

[23]: <Axes: xlabel='quality', ylabel='free sulfur dioxide'>

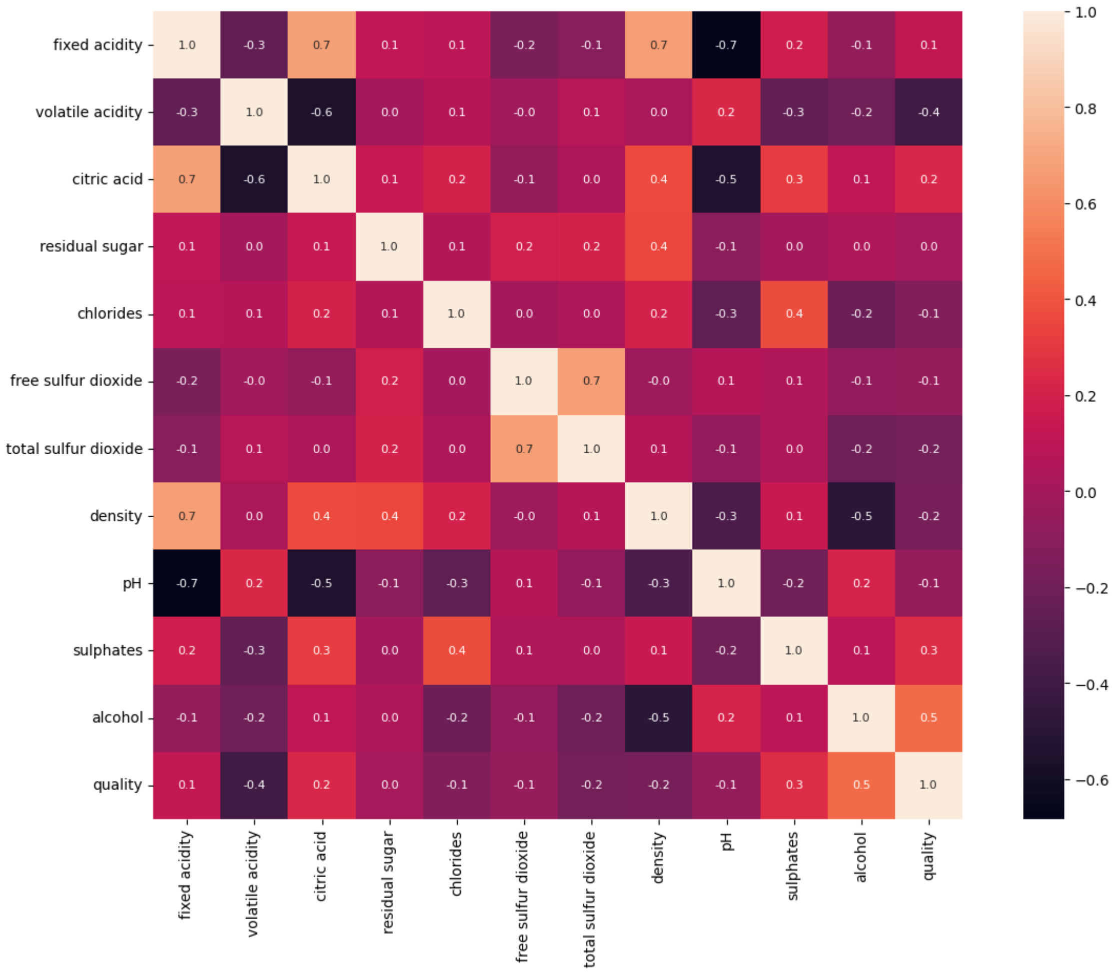


Correlation

[24]:
```
Corr = df.corr()
```

[25]:
```
plt.figure(figsize=(15,10))
sns.heatmap(Corr, cbar=True, square=True, fmt = '.1f', annot = True,␣
  ↪annot_kws={'size':8})
```

[25]: `<Axes: >`



Data Preprocessing

Separate the Data and label

[26]:
```python
X = df.drop('quality',axis=1)
```

[27]:
```python
print(X)
```

```
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0               7.4             0.700         0.00             1.9      0.076
1               7.8             0.880         0.00             2.6      0.098
2               7.8             0.760         0.04             2.3      0.092
3              11.2             0.280         0.56             1.9      0.075
4               7.4             0.700         0.00             1.9      0.076
…                …                 …            …               …          …
1594            6.2             0.600         0.08             2.0      0.090
1595            5.9             0.550         0.10             2.2      0.062
1596            6.3             0.510         0.13             2.3      0.076
```

```
1597             5.9             0.645         0.12             2.0       0.075
1598             6.0             0.310         0.47             3.6       0.067

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                    11.0                  34.0  0.99780  3.51       0.56
1                    25.0                  67.0  0.99680  3.20       0.68
2                    15.0                  54.0  0.99700  3.26       0.65
3                    17.0                  60.0  0.99800  3.16       0.58
4                    11.0                  34.0  0.99780  3.51       0.56
…                     …                   …       …      …        …
1594                 32.0                  44.0  0.99490  3.45       0.58
1595                 39.0                  51.0  0.99512  3.52       0.76
1596                 29.0                  40.0  0.99574  3.42       0.75
1597                 32.0                  44.0  0.99547  3.57       0.71
1598                 18.0                  42.0  0.99549  3.39       0.66

      alcohol
0         9.4
1         9.8
2         9.8
3         9.8
4         9.4
…         …
1594     10.5
1595     11.2
1596     11.0
1597     10.2
1598     11.0

[1599 rows x 11 columns]
```

[28]:
```python
Y = df['quality'].apply(lambda y_value: 1 if y_value>=7 else 0)
```

[29]:
```python
print(Y)
```

```
0       0
1       0
2       0
3       0
4       0
       ..
1594    0
1595    0
1596    0
1597    0
1598    0
Name: quality, Length: 1599, dtype: int64
```

Train Test Split

[30]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2,␣
↪random_state=3)
```

Model Training

[31]:
```
model = RandomForestClassifier()
```

[32]:
```
model.fit(X_train, Y_train)
```

[32]:
```
RandomForestClassifier()
```

Evaluating The Model

Accuracy on test data

[33]:
```
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

[34]:
```
print('Accuracy : ', test_data_accuracy)
```

```
Accuracy :  0.928125
```

[35]:
```
input_data = (7.5,0.5,0.36,6.1,0.071,17.0,102.0,0.9978,3.35,0.8,10.5)
```

Changing the input data to a numpy array

[38]:
```
input_data_to_numpy_array = np.asarray(input_data)
```

[39]:
```
input_data_reshape = input_data_to_numpy_array.reshape(1,-1)
#Here we are reshaping the data as we are predicting the label for only one␣
↪instance
```

Prediction

[40]:
```
prediction = model.predict(input_data_reshape)
print(prediction)

if (prediction[0]==1):
  print('Good Quality Wine')
else:
  print('Bad Quality Wine')
```

```
[0]
Bad Quality Wine
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but RandomForestClassifier was fitted with feature
```

```
names
  warnings.warn(
```

[ ]:

```

```